CSC 373: Algorithm Design and Analysis
Solutions for questions 6 and 7 in problem set 1

Allan Borodin

February 1, 2013
Consider the following variant of the knapsack problem. There are \( n \) items and each item can be taken \( j \) times for \( j \in \{0, 1, 3\} \) times. Assume further that the size of each item is an integer \( \leq n^2 \). Provide a dynamic programming (DP) algorithm for this problem.

Solution: It wasn’t stated but assumed that there is a knapsack bound \( B \).

- Provide a semantic array definition for computing the cost of an optimal solution.

As in the standard knapsack, we will define the semantic array:

\[
V[i, b] = \text{the maximum profit possible using only the first } i \text{ items and not exceeding the bound } b.
\]

\[
0 \leq i \leq n; \ 0 \leq b \leq B.
\]

We can then assume \( B \leq n^3 \).
Solution for problem 6 continued

- Provide a recursive (computational) definition for computing values of this array and briefly justify why your computational definition is equivalent to the semantic definition.

The corresponding computational array is:

\[
V'[i, b] = \begin{cases} 
0 & \text{if } i = 0 \text{ or } b = 0 \\
\max\{C, D, E\} & \text{if } s_i \leq b
\end{cases}
\]

where

\[
C = V'[i-1, b], \quad D = V'[i-1, b-s_i] + v_i \quad \text{and} \quad E = V'[i-1, b-3s_i] + 3v_i.
\]

The three cases in the computational array correspond (respectively) to the cases when the maximum profit defining \(V[i, b]\) does not use the \(i^{th}\) item (resp. uses on copy, uses three copies of the \(i^{th}\) item).

- What is the asymptotic complexity of your algorithm in terms of the number of arithmetic operations and comparisons as a function of \(n\).

The size of the array is at most \((n + 1)B = O(n^4)\) and each entry of \(V'\) requires \(O(1)\) to compute given previous entries.
Solution for question 7

It wasn’t stated but I meant this question to count 15 points. I will provide a semantic array, and a computational array for the problem as stated in question 6.27 of the text. This is a maximization problem but there are also penalties involved. The text relates the question to 6.26 where the “penalty” for matching with a “-” is expressed as $\delta(-, y)$ and $\delta(x, -)$ where the natural interpretation would be that these would be negative. For 6.27 we will just subtract $c_0 + c_1 k$ whenever there is a gap of length $k$ being inserted.

- The semantic array will be $V[i, j] = (s, g)$ where $s$ is the maximum score for matching the strings $x[1...i]$ and $y[1...j]$ for $0 \leq i \leq n$ and $0 \leq j \leq m$ and $g$ is an indicator defined so that $g = 0$ means that $x(i)$ and $y(j)$ are being matched, $g = 1$ means that $x(i)$ is being matched with an inserted gap and $g = 2$ means that $y(j)$ is being matched with an inserted gap.
The computational array will be

\[
V'[i, b] = \begin{cases} 
0 & \text{if } i = j = 0 \\
c_0 + c_1j & \text{if } i = 0 \text{ and } j > 0 \\
c_0 + c_1i & \text{if } i > 0 \text{ and } j = 0 \\
\max\{(C, 0), (D, 1), (E, 2)\} & \text{if } i > 0 \text{ and } j > 0 
\end{cases}
\]

where

\[
C = V'[i - 1, j - 1] + \delta(x_i, y_j)
\]

\[
D = V'[i - 1, j] - [c_0 + c_1] \text{ if } V'[i - 1, j] = (s, 0) \text{ for some } s; \\
= V'[i - 1, j] - c_1 \text{ if } V'[i - 1, j] = (s, 1) \text{ for some } s
\]

\[
E = V'[i, j - 1] - [c_0 + c_1] \text{ if } V'[i, j - 1] = (s, 0) \text{ for some } s; \\
= V'[i, j - 1] - c_1 \text{ if } V'[i, j - 1] = (s, 2) \text{ for some } s
\]

There are \((n + 1)(m + 1)\) entries and each entry takes \(O(1)\) to compute given previous entries so that the complexity is \(O(mn)\).